A Holistic Approach for Enhancing Data Integrity and Reliability in Human-Robot Interaction

Manuel Meder*
University of Bremen
Bremen, Germany
mmeder@uni-bremen.de

Kashmira Shinde*
University of Bremen
Bremen, Germany
kshinde@uni-bremen.de

Dennis Hemker*
University of Bremen
Bremen, Germany
hemker@uni-bremen.de

Sadique Siddiqui
University of Bremen
Bremen, Germany
siddiqui@uni-bremen.de

Teena Hassan
University of Bremen
Bremen, Germany
thassan@uni-bremen.de

Nina Hoyer
University of Bremen
Bremen, Germany
nihoyer@uni-bremen.de
DFKI, Robotics Innovation Center
Bremen, Germany
nina.hoyer@dfki.de

Elsa Andrea Kirchner
DFKI, Robotics Innovation Center
Bremen, Germany
elsa.kirchner@dfki.de
University of Duisburg-Essen
Duisburg, Germany
elsa.kirchner@uni-due.de

ABSTRACT

Space is a remote and hostile environment, and it poses several challenges to both, humans and robots deployed in space missions. This necessitates the use of robotic systems on which humans can rely on. However, in extreme environments, sensors might fail or deliver incorrect measurements. Furthermore, signal noise and model inaccuracies could result in inconsistencies between information extracted using different sensors or algorithms. Erroneous and unreliable data can cause autonomous systems to behave in unexpected or inappropriate ways, which reduces human trust in such systems. This in turn would hamper the assistive potential of robots in space missions. Methods that identify and handle sensor failures and low-level data inconsistencies are commonly applied to strengthen the system's robustness. However, few have addressed logical inconsistencies that occur at higher semantic levels of data processing. By following a holistic approach that checks for, and handles both types of inconsistencies, we can enhance the integrity of the data that is used for higher-level inferences and decision-making by robots. This would make robots a more reliable interaction partner for humans. Therefore, we propose an architecture to detect and respond to inconsistencies at both the local and the global levels of data processing. A first implementation of the proposed approach includes auxiliary methods to perform preliminary checks and to facilitate integration into a robot's dataflow architecture. The proposed architecture could also be applied to terrestrial use cases (e.g., in healthcare), where data integrity, reliability, and trust play a crucial role.

CCS CONCEPTS

• **Computer systems organization** → **Reliability**; *Robotic components.*

KEYWORDS

Trust, Data Integrity, Architecture, Consistency, HRI

ACM Reference Format:
Manuel Meder, Kashmira Shinde, Dennis Hemker, Sadique Siddiqui, Teena Hassan, Nina Hoyer, and Elsa Andrea Kirchner. 2022. A Holistic Approach for Enhancing Data Integrity and Reliability in Human-Robot Interaction. In *SpaceCHI 2.0 workshop at ACM SIGCHI 2022*. New Orleans, LA, USA, 5 pages.

1 INTRODUCTION

The desire to better understand the universe has fueled human endeavors in reaching and exploring space environments - be it in-orbit, near-Earth or deep space. Such an environment has large uncertainties and is usually unpredictable. Reliance on robotic assistance in such a scenario not only improves robustness and productivity, but also enables a wider variety of space missions, including future manned missions. The more complex the interaction between a human and a robotic system gets, the more sensor data must be processed. This in turn leads to a higher probability of errors and inconsistencies in sensor data and models which may affect the

*Authors contributed equally to this research.

SpaceCHI 2.0, May 01, 2022, New Orleans, LA

decision-making capabilities of the robot, thereby making it less reliable.

Reliability and trust in an autonomous system are interrelated [4]. The work presented in [5] defines trust in autonomous systems as "...an attitude which includes the belief that the collaborator will perform as expected, and can, within the limits of the designer's intentions, be relied on to achieve the design goal". A decrease in reliability can lead to a decrease in system trust [4].

To stay operational with the best possible accuracy and reliability, inconsistencies in sensor data and model predictions need to be first detected and then also handled. With inconsistencies we refer to occurrences fitting the following description:

- A raw sensor value that violates preliminary assumptions of its operational range. These assumptions comprise a range of values, sampling rates and data evolution over time.
- A logical proposition that doesn't fit the known representation of the world. This includes the simultaneous occurrence of physically exclusive sensor data processing results.

Prior research has proposed different techniques to detect and handle sensing failures. Sensor fault detection can be done using model-based methods [3, 10]. Wu et al. [11] have shown that, through the combination of model-based methods and rule-based methods, systems can also check for logical implications. Another approach is to perform a correlation test on the readings of different sensors for determining redundant information, anomalies or sensor faults [2, 3]. Qu and Veres [9] focus in their work on the detection of logical consistency checks for robotic agents. Their modelling formalism allows them to detect inconsistencies regarding rules, the robot's state belief and actions for real-time applications. In [6], an exception handling mechanism for the classification of sensor failures and its recovery by functionally replacing the fault causing sensor with an alternate sensor has been described. This mechanism has a global scope which enables it to find an appropriate replacement from the sensors allocated to other behaviors. A global behavioral or task monitor is responsible for detecting a sensor fault and forwarding the information about the source of the error to the exception handler. Furthermore, they implement an interface to the behavioral component of an autonomous system to also act upon detected inconsistencies.

While all the above-mentioned works cover different key aspects for ensuring data integrity, a general architecture to integrate the different methods is lacking. In this paper, we present a holistic architecture which aims to combine the approaches of previous work to detect inconsistencies at different levels of the data processing chain (Sec. 2). Our contribution is a concept to detect and handle inconsistencies in local information (Sec. 2.3) as well as logical contradictions that may arise when combining data from various processing nodes (Sec. 2.4). Checks constrained to local information from a data processing node are performed as early as possible to ensure the removal of unreliable data from the processing pipeline, resulting in a more efficient use of computational resources. Checks for inconsistencies are performed in parallel by combining information from multiple sources comprising live data and stored information (e.g., database, expert knowledge). The proposed architecture is currently being realized as part of the KiMMI SF project [7] (Sec. 3.1). We briefly describe the current implementation of the consistency management (Sec. 3.2) and give an outlook on subsequent steps and challenges (Sec. 3.3) that need to be tackled.

2 CONSISTENCY MANAGEMENT ARCHITECTURE

In this section, we provide an overview of our proposed architecture for detecting and handling local and global inconsistencies.

2.1 General communication layers

Within a robotic system, several communication layers exist to create task-specific abstraction and yield via their interplay a complex behavior and thus a high level of autonomy. Layers relevant to this work, including the *Consistency Management*, are displayed in Figure 1. *Consistency Management* resides on the same conceptual level as the *Dataflow Manager*. With the help of the *Consistency Management*, the *Dataflow Manager* restructures the information flow between processing nodes to incorporate consistency checks. By doing so, processing nodes themselves as well as high-level modules access the output of processing nodes via the *Consistency Management*.

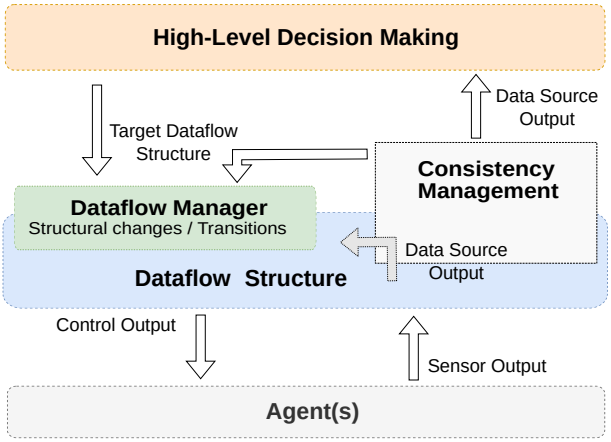

Figure 1: A schematic abstraction of communication layers for a robotic system is shown. The consistency management resides on the level of the data flow. It has interfaces to the data flow management to gain access to data source outputs. Connection in the other direction can be used to introduce adaptations of processing nodes via the data flow management. To enforce valid data the components on the higher level decision making also access the computed information through the consistency check module.

2.2 Requirements for consistency management

An architecture that is capable of detecting and handling data inconsistencies needs to fulfill multiple requirements, four of which we consider here. In complex robotic systems, a vast amount of data producing, and data processing nodes are integrated. The task at hand to always maintain a reliable state of the entire system is a challenge that can be partitioned structurally into two scopes.

Within this architecture, we call those the *local inconsistencies* and the *global inconsistencies*. The former concentrates on inconsistencies that occur, can be detected and handled within a single data processing node **(REQ1)**. The global scope targets those inconsistencies at a higher semantic level, which can only be detected when considering data from multiple processing nodes in conjunction **(REQ2)**. As computational resources on mobile platforms are precious, the architecture should limit dispensable data processing early on in cases of poor data quality **(REQ3)**. Another requirement is to communicate information about the detected inconsistencies to other software components **(REQ4)**, which can also be forwarded to the human interaction partner, thus strengthening the trust in the decisions of autonomous robots.

2.3 Local inconsistencies

On the scope of *local inconsistencies*, we consider data anomalies that violate predefined checking mechanisms. All the information needed to detect such anomalies is present at the level of the data processing node itself. This does not exclude additional information obtained from a centralized knowledge storage. We distinguish here between three categories of *local inconsistencies*:

- **Missing data:** Due to sensor faults and external factors, sensor data can be missing partially or also completely. Algorithms can fail to calculate output data. Missing data can lead to whole processing chains not working correctly and therefore need to be detected to take countermeasures by using available redundancy or safety limitations of the robot's actions.
- **Invalid data:** Available data can still be invalid. E.g., the value is outside of the working range or NaN (not a number) values are delivered. Data from stuck sensors can also be considered invalid. Subsequent processing nodes must be made aware of this to be able to handle the data accordingly.
- **Inaccurate data:** Due to noise, sensor inaccuracies and bad calibrations as well as algorithmic inaccuracies, data can only be trusted to a certain extent. Subsequent processing nodes must be made aware when dealing with inaccurate input data.

To handle *local inconsistencies*, our proposed architecture manipulates the dataflow of individual processing nodes **(REQ1)** as depicted in Figure 2. Pre-regulators and post-regulators wrapped around the actual processing node enable the consistency module to check for the above-mentioned *local inconsistencies*. Invalid data is detected early on and thus fulfilling the requirement to save computational resources by preventing useless calculations **(REQ3)**. Data produced by a node is checked and annotated accordingly for consumption by other nodes. The local *Consistency Module* itself has access to global stored knowledge obtained by the system and additional required predefined knowledge which can be stored, for example as knowledge rules. Furthermore, a list of crucial inputs for the actual processing node is required. In systems with input modules calculating redundant information, this list is used for decision-making on when to prevent calculation or label the output as invalid or inaccurate.

Many methods exist for detecting *local inconsistencies*. Methods can be signal-based, statistics-based, model-based, rule-based and many more. The authors of [8] give a good overview of available algorithms that can be incorporated as checking mechanisms within this architecture.

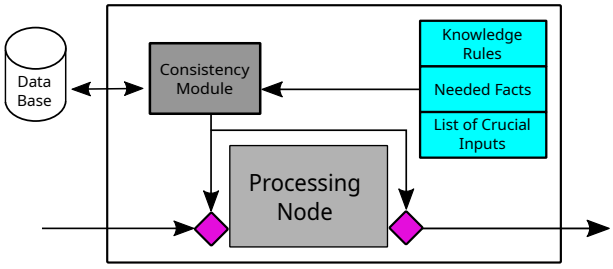

Figure 2: Depiction of the consistency check on the local scope. Regulators shown as diamonds before the entry and exit point of the actual processing node serve as interaction points for the checking methods. By this design, the processing node is not aware of its surroundings and can expect valid data. The consistency module additionally has access to global stored knowledge such as databases as well as knowledge rules and required facts. A list of crucial inputs aids the consistency module to suppress the waste of computation time when required input is missing or invalid.

2.4 Global inconsistencies

In addition to the *local inconsistencies*, we also introduce the concept of *global inconsistencies*. A component observing and checking the output of multiple nodes is called a *Global Monitor* **(REQ2)**. It makes use of "knowledge rules" and "levels of trust" and is depicted in Figure 3. Knowledge rules would represent known or learned concepts about the world (e.g., physical constraints) that can be used to inspect the combined output of nodes. Levels of trust enable the *Global Monitor* to emphasize the output of an incoming processing node. It can be used to tell other nodes that in case of "equally inaccurate data" of sensors A and B they should stick to e.g., sensor A's values. As an example, suppose a node A detects the human's position in a specific reference frame in meters like $pos_A(human) = \begin{bmatrix}12.5 & 40.0 & 1.2\end{bmatrix}$. In contrast, a node B provides a position calculation $pos_B(human) = \begin{bmatrix}12.5 & 45.0 & 1.2\end{bmatrix}$ using another algorithm. This causes a physical constraint violation as the same human cannot be in two positions at the same time with a $\Delta y = 5m$. Outgoing data is then annotated respectively or suppressed (Sec. 2.5).

By design, it is possible that multiple checks within a *Global Monitor* fail at the same time. This depends on the chosen algorithms, their combination and processing sequence. *Global Monitors* between different nodes can only be set up if these nodes share any properties that can be exploited.

2.5 Handling of inconsistencies

In a complex human-robot interaction scenario, the detected inconsistencies can be of varying importance. While small changes in a sensor's accuracy might not negatively affect a classification algorithm at all, stronger deviances can lead to unexpected system

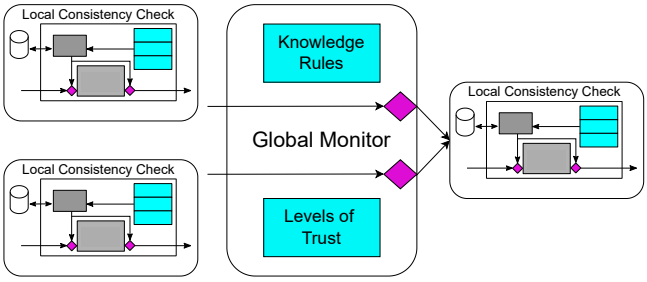

Figure 3: Illustration of a global monitor. They are located between processing nodes and inspect the data utilizing knowledge about common properties. Regulators shown as diamonds again can suppress or annotate outgoing data. Knowledge rules make the monitor aware of e.g., physical constraints and levels of trust define the weighting of incoming processing node information.

behavior. Such situations can even be highly critical for the collaborating human as the robot could move its limbs very quickly to wrong positions. Therefore, we introduce an *action layer* to react on multiple levels with suitable impact.

Actions can be divided into two groups - *local* and *global*. Each has a varying impact on the system. *Local actions* include direct annotation of the data to inform other consumers in the network about inconsistencies **(REQ4)**. Furthermore, textual explanations might help a human operator to analyze the situation **(REQ4)**. As *local actions* with medium impact, we propose to use techniques that replace inconsistent data or missing values if possible (cf. [1]) or to pause data processing temporarily. While these *actions* only affect a node itself, *global actions* influence the whole topology. Such actions request decision-making modules to turn nodes off completely or to spawn a replacement. Additionally, if measurements become too unreliable, requests for suppression of the current behavior up to a complete system halt can be made. We suggest seeing these *global actions* as notifications to an external component that takes care of further steps.

3 SOFTWARE DEVELOPMENT

This section presents an overview of the software framework development project and the current implementation of the proposed consistency management architecture within this framework.

3.1 KiMMI Software Framework

KiMMI SF aims at realizing a software framework for flexible human-machine interaction that is appropriate for the respective context. This adaptive software framework supports online reconfigurable dataflow and integrates models, basic components and interfaces for continuous determination and evaluation of the context (the situation) as well as user's intention for situation-aware assistance, while ensuring data integrity.

Our proposed architecture for consistency management will be integrated to detect inconsistencies at different parts of the framework and will be evaluated on a real robot within the project. While KiMMI SF is related to space, the architecture can also be applied to terrestrial use cases.

3.2 Implementation

The proposed architecture is currently developed as a Python module. With the help of module extensions, writing checking methods in C++ is also supported. Within a first proof of concept demonstration, *local inconsistencies* were checked with the help of signal-based methods. In a reduced functional framework, the consistency check was integrated and helped to examine a fixed sequence of images from a virtual sensor. The consistency checks annotated data accordingly and stopped propagation early when images were too dark or too bright for subsequent calculations of detected human and object poses. By doing so, the requirement to save processing resources **(REQ4)** was satisfied. The specific implementation interfaces with other components of the robotic framework to gain access to the processing nodes and their outputs as well as to communicate suggested changes in the dataflow. All of this is abstracted away from the end user where possible. Given a description of the desired processing node network, the *Consistency Management* offers functionality to automate the creation of the modified network involving the consistency checks. This new dataflow structure can then be consumed by the *Dataflow Manager* to execute the required adaptations. This reduces the involvement of human experts when applying the robotic software framework.

3.3 Future work

The deployment of checks for *local inconsistencies* can be further automated, if the dataflow management offers an interface for instantiation of processing nodes from a description file. To provide even more assistance, a library of common checking methods will be provided with the final framework. Where this library does not suffice the needs due to the nature of the specifics of the processing nodes and their tasks, interfaces will be provided to allow the integration of additional custom-written checking methods, thereby extending the capabilities of the *Consistency Management*. Further work will include the implementation of the *Global Monitors*. Interesting challenges reside in the automatic deployment of such monitors from description files including knowledge rules that must be checked. With more complex scenarios that are planned within the project, it will be interesting to see how well the monitors scale with increasing number of processing nodes that have dependencies due to given rules. Time performance evaluations will be conducted to gain insights on the efficiency of consistency management on a robotic system. Consistency checks can be temporally and selectively switched off if desired by the decision-making components or the humans. It will be interesting to investigate how consistency management in general and the ability to switch it off by the human affect trust in the system.

ACKNOWLEDGMENTS

This work was supported through a grant of the German Federal Ministry of Economic Affairs and Climate Action (BMWK, FKZ 50 RA 2022).

REFERENCES

[1] Miquel À. Cugueró-Escofet, Diego García, Joseba Quevedo, Vicenç Puig, Santiago Espin, and Jaume Roquet. 2016. A methodology and a software tool for sensor data validation/reconstruction: Application to the Catalonia regional water network. 49 (04 2016), 159–172. https://doi.org/10.1016/j.conengprac.2015.11.005

[2] Arun Ganesan, Jayanthi Rao, and Kang Shin. 2017. Exploiting Consistency Among Heterogeneous Sensors for Vehicle Anomaly Detection. https://doi.org/10.4271/2017-01-1654 ISSN: 0148-7191, 2688-3627.

[3] Eliahu Khalastchi, Meir Kalech, and Lior Rokach. 2013. Sensor Fault Detection and Diagnosis for Autonomous Systems. In *Proceedings of the 2013 International Conference on Autonomous Agents and Multi-Agent Systems* (St. Paul, MN, USA) *(AAMAS '13)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 15–22.

[4] Michael Lewis, Katia Sycara, and Phillip Walker. 2018. Trust in Human-Robot Interaction. In *Foundations of trusted autonomy*, Hussein A. Abbass, Jason Scholz, and Darryn J. Reid (Eds.). Springer, Cham, 135–159. https://doi.org/10.1007/978-3-319-64816-3

[5] Neville Moray and T. Inagaki. 1999. Laboratory studies of trust between humans and machines in automated systems. *Transactions of the Institute of Measurement and Control* 21, 4-5 (1999), 203–211. https://doi.org/10.1177/014233129902100408

[6] Robin R. Murphy and David Hershberger. 1996. Classifying and Recovering from Sensing Failures in Autonomous Mobile Robots. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence - Volume 2* (Portland, Oregon) *(AAAI'96)*. AAAI Press, 922–929.

[7] University of Bremen and German Research Center for Artificial Intelligence. 2020. KiMMI SF - Adaptive software framework for context-sensitive, intuitive man-machine-interaction. https://robotik.dfki-bremen.de/en/research/projects/kimmi-sf/. Accessed: 2022-03-04.

[8] Ivan Miguel Pires, Nuno M. Garcia, Nuno Pombo, Francisco Flórez-Revuelta, and Natalia Díaz Rodríguez. 2016. *Validation Techniques for Sensor Data in Mobile Health Applications.* https://doi.org/10.1155/2016/2839372 ISSN: 1687-725X Pages: e2839372 Publisher: Hindawi Volume: 2016.

[9] Hongyang Qu and Sandor M. Veres. 2016. Verification of logical consistency in robotic reasoning. *Robotics and Autonomous Systems* 83 (2016), 44–56. https://doi.org/10.1016/j.robot.2016.06.005

[10] Tim Tiedemann, Elmar Berghöfer, Christian Rauch, and Frank Kirchner. 2014. Sensor Fault Detection and Compensation in Lunar/Planetary Robot Missions Using Time-Series Prediction Based on Machine Learning. *Acta Futura* 9 (01 2014), 9–20. https://doi.org/10.2420/AF09.2014.9

[11] Li-hua Wu, Yun-fei Jiang, Wei Huang, Ai-xiang Chen, and Xue-nong Zhang. 2006. The Intelligent Fault Diagnosis for Composite Systems Based on Machine Learning. In *2006 International Conference on Machine Learning and Cybernetics*. 571–575. https://doi.org/10.1109/ICMLC.2006.258337 ISSN: 2160-1348.